Secure by Design vs Secure by Default

"Secure by Design" and "Secure by Default".
Two terms often used in the same breath and interchangeably.
Meaning very different things.

Secure by Design is about how you build.

It means security is considered from the start, not bolted on after the architecture is set. Threat modeling happens before design decisions are finalized. Security requirements are derived from risk analysis, not added as a checklist at the end of the sprint. The development process itself is structured to prevent classes of vulnerabilities from being introduced in the first place.

Secure by Default is about what ships.

It means the product, out of the box, is configured in the most secure way that is practical for the intended use case. No unnecessary services enabled. Authentication required. Logging on. Passwords not "admin/admin".

A product can be built with Secure by Design principles and still ship with insecure defaults.
And a product with locked-down defaults can still be full of architectural vulnerabilities.

You need both.
They address different problems at different points in the lifecycle.
Confusing them creates blind spots.

Secure by Design prevents vulnerabilities.
Secure by Default prevents unnecessary exposure.

The EU Cyber Resilience Act requires both. Not as a philosophical exercise, but as verifiable obligations.
That distinction matters when you're trying to demonstrate compliance, not just claim it.

In practice, the design gap is usually the harder problem.
Insecure defaults are embarrassing.
Security architecture mistakes are expensive.

And much harder to fix once the product ships.